**CHAPTER 1**

**INTRODUCTION**

* 1. **INTRODUCTION TO DBMS**

Database is a collection of related data and data is collection of facts and figures that can be procedure information

Mostly data represents recordable facts. Data aids in procedure information, which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks. A Database management system stores data in such a way that it become easier to retrieve, manipulate, and procedure information.

**Characteristics of Database Management System :**

Traditionally, data was organized in file formats. DBMS was a new concept then, and all the research was done to make it overcome the deficiencies in traditional style of data management. A modern DBMS has the following Characteristics.

• Real-word entity

• Relation-based tables

• Isolation of data and application

• Less redundancy

• Consistency

• Query language

• ACID Properties

• Multiple views

• Multi users and concurrent access

• Security

**Advantages and Disadvantages of DBMS:**

Advantages:

• Reduction of Redundancy :This is a perhaps the most significant advantages of using DBMS.

Redundancy creates several problems like, requiring extra storage space, entering same data more than once during data insertion, and deletion data from more than one place during deletion.

Sharing of data :In paper-based record keeping, data cannot be shared among many users, But in DBMS, many users can share the same database if they connected via network.

Disadvantages:

As DBMS needs computer, we have to invest a good amount in acquiring the hardware, software, installation facilities and training of user. We need to keep regular backups because a failure can occur any time. Taking backup is a lengthy process and the computer system cannot perform any other jobs at this times.

* 1. **ABOUT SQL**

SQL is a language to operate database; it includes database creation, deletion, fetching rows, modifying rows, etc. SQL is an ANSI (American National Standards Institute) standard language, but there are many different versions of SQL language.

**What is SQL :**

SQL is a Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relation database. SQL is the standard language for Relational Database System. All the Relation Database System. All the Relation Database Management System (RDBMS) like MYSQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

**Why SQL ?**

SQL is widely popular because it offers the following advantages :

• Allows user to access data in the RDBMS.

• Allows the user to describe the data.

• Allows user to create and drop database and table.

• Allows user to set permission on table, procedures and views.

• Allows user to define data in a database and manipulate that data.

• Allows to embed within other language using SQL modules, libraries and pre-compilers.

**1.3 PROJECT DESCRIPTION**

The main objective of this project is to provide communication within an organisation. It provides facility for students in an organization to interact and share ideas. This project makes two persons anywhere within the campus communicate from any local system connected to the server. Keeping in mind that students’ interaction can make the students bright and strong in circular and other activities, we developed this software STUDENT MESSENGER.

Not in many educational institutions, there is a communication medium for students. Our projects is a solution for this problem. So, students can always communicate with each other using this system for free of cost instead of wasting money for mobile bills.

The following are the benefits provided by the system:

* Saves more time.
* Students can exchange their views using forums.
* Can stay in touch with friends.
* Provides security by giving permissions separately to administrator and user.
* Any user can easily access the forums questions and answers.
* The users can be deleted by the administrator if an user is offensive.

**1.4 PROBLEM SPECIFICATION:**

The project “STUDENT MESSENGER” aims at communication within a local area network. It provides facility for students in an organization to interact and share ideas. This project makes two persons anywhere within the campus communicate from any local system connected to the server. It also deals with maintenance of discipline of the communication.

The following are the main facilities offered by the system:

* Generating list of users registered.
* Registration of new users.
* Sending and receiving mails.
* Online chat between users.
* Maintaining personal contacts.
* Posting a question in forums by users.

**CHAPTER 2**

**REQUIREMENTS SPECIFICATION**

**2.1 SOFTWARE REQUIREMENTS:**

* Operating system : WINDOWS XP or RED HAT or LINUX
* Connectivity : PHP
* Data base : MYSQL
* Web browser : MOZILLA or IE
* Front end : HTML

**2.2 HARDWARE REQUIREMENTS:**

* Processor : PENTIUM-IV & above.
* RAM : 256MB & above.
* Hard Disk : 20GB & above.
* Monitor

**Chapter 3**

**DESIGN**

**3.1 Schema Diagram:**

**users**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| userid | password | fname | lname | address | state | country | zipcode | branch | sex | dob |

**reports**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| student\_id | reason\_text | reporter\_id | rid | subject |

**history**

|  |  |  |  |
| --- | --- | --- | --- |
| id | user\_id | action | login\_info |

**messages**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| msg\_id | receiver\_id | sender\_id | msg\_read | msg\_sub | msg\_time | msg\_body |

**Contacts**

|  |  |
| --- | --- |
| user\_id | contact\_id |

|  |  |  |  |
| --- | --- | --- | --- |
| sender\_id | received\_id | msg | msgtime |

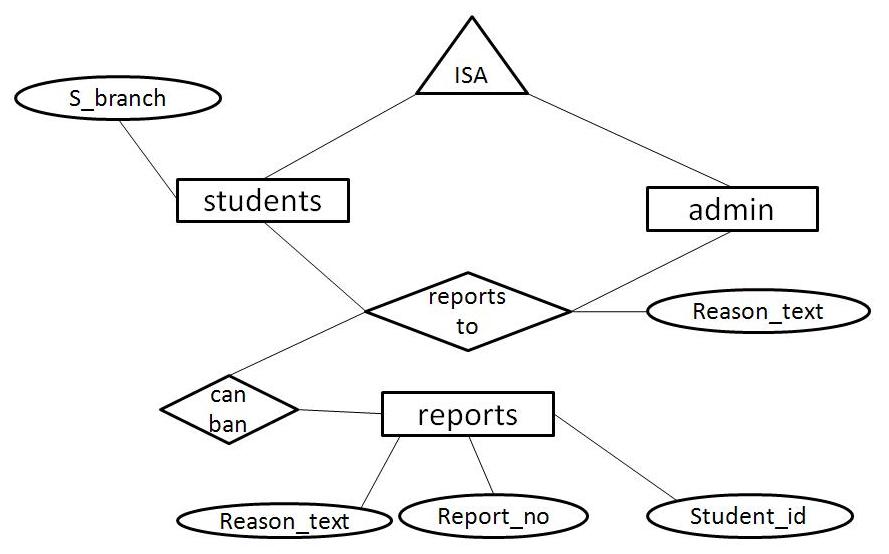
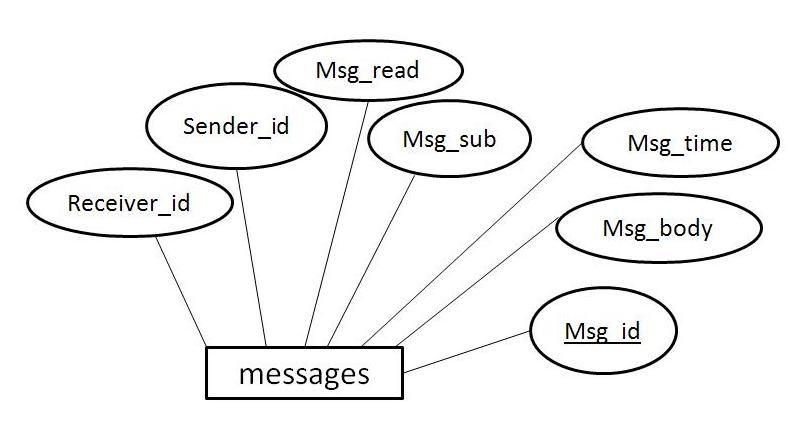
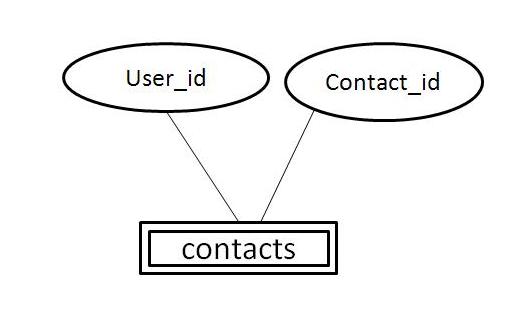
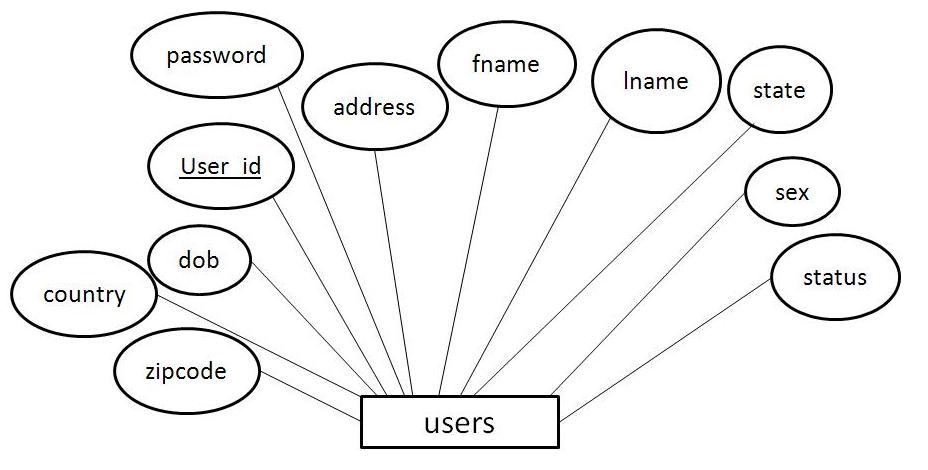
**Chat**

**Admin**

|  |  |
| --- | --- |
| username | password |

**3.2 E-R DIAGRAM:**

Entity Relationship model (ER model for short) is an abstract way to describe a database design. It usually starts with a relational database, which stores data in tables. Some of the data in these tables point to data in other tables.Data in the table is retrieved using primary key. The primary key is the one which uniquely identifies each record in the table in relational database management system.In ER Diagram an entity may be defined as a thing which is recognized as being capable of an independent existence and which can be uniquely identified. An entity is an abstraction from the complexities of a domain.A relationship captures how entities are related to one another. Entity–relationship diagrams don't show single entities or single instances of relations. Rather, they show entity sets and relationship sets.In software engineering an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model that defines a data or information structure that can be implemented in a database, typically a relational database.Entity–relationship modeling was developed for database design by Peter Chen and published in a 1976 paper. However, variants of the idea existed previously.



**Can post**

Friendsnda

n

1

n

1

**3.3 DATA FLOW DIAGRAM (DFD):**

A data flow diagram(DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangle, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually „say‟ things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO.

Notations of DFD:

Functions

file/database

Input/Output

USER

Fig:- DATA FLOW DIAGRAM

**3.4 Use case Diagram**

Use-case diagram is a coherent piece of functionality that a system can provide by interacting with actors. In our system, all the modules are going to interact with one or the other actors (different actors could be, registered users, new users, database admin). The use case technique is used to capture a system behavioral by dealing scenario drives threads through the functional requirements. The components in a use case diagram include:

* **Use cases:** A use case describes a sequence of actions that provides something of measurable value to an actor and is drawn as a horizontal ellipse.
* **Actors:** An actor is a person, organization, or an external system that plays a role in one or more interactions with the system.
* **Associations:** Associations are shown between actors and use cases by drawing a solid line between them.

ADMIN

USER USER

Fig:- Use case diagram

**CHAPTER 4**

**Implementation**

**4.1 Description of Tables**

**CREATE TABLE USERS(**

USER\_ID VARCHAR(15) NOT NULL,

PASSWORD VARCHAR(15) NOT NULL,

FNAME VARCHAR(15) NOT NULL,

LNAME VARCHAR(15) NOT NULL,

ADDRESS VARCHAR(30),

STATE VARCHAR(20),

COUNTRY VARCHAR(20),

ZIPCODE INTEGER DEFAULT 0,

S\_BRANCH VARCHAR(5) NOT NULL,

SEX VARCHAR(6) NOT NULL,

DOB DATE NOT NULL,

STATUS INTEGER DEFAULT 0,

PRIMARY KEY (USER\_ID));

**CREATE TABLE ADMIN(**

USERNAME VARCHAR(15) NOT NULL,

PASSWORD VARCHAR(15) NOT NULL,

PRIMARY KEY(USERNAME));

**CREATE TABLE CONTACTS(**

**USER\_ID** VARCHAR(15),

CONTACT\_ID VARCHAR(15));

**CREATE TABLE CONTACTS**

MSG\_ID INTEGER AUTOINCREMENT,

RECEIVER\_ID VARCHAR(15) NOT NULL,

SENDER\_ID VARCHAR(15) NOT NULL,

MSG\_READ TINYINT DEFAULT 0,

MSG\_SUB VARCHAR(100),

MSG\_TIME TIMESTAMP,

MSG\_BODY BLOB,

PRIMARY KEY(MSG\_ID),

FOREIGN KEY(RECEIVER\_ID) REFERENCES USERS(USER\_ID),

FOREIGN KEY(SENDER\_ID) REFERENCES USERS(USER\_ID);

**CREATE TABLE CHAT**

SENDER\_ID VARCHAR(15) NOT NULL,

RECEIVER\_ID VARCHAR(15) NOT NULL,

MSG VARCHAR(200),

**CREATE TABLE REPORTS(**

STUDENT\_ID VARCHAR(15),

REASON\_TEXT TEXT,

REPORTER\_ID VARCHAR(15).

RID INTEGER AUTO INCREMENT,

SUBJECT VARCHAR(50),

PRIMARY KEY(RID),

FOREIGN KEY(STUDENT\_ID) REFERENCES USERS(USER\_ID)

FOREIGN KEY(REPORTER\_ID) REFERENCES USERS(USER\_ID));

**CREATE TABLE HISTORY(**

ID INT(20) NOT NULL,

UESER\_ID VARCHAR(20),

ACTION VARCHAR(20),

LOGIN\_INFO DATETIME);

**Users table:-**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| user\_id | varchar(15) | NO | PRI | NULL |  |
| Password | varchar(15) | NO |  | NULL |  |
| Fname | varchar(15) | NO |  | NULL |  |
| lname | varchar(15) | NO |  | NULL |  |
| Address | varchar(15) | YES |  | NULL |  |
| State | varchar(15) | YES |  | NULL |  |
| country | varchar(15) | YES |  | NULL |  |
| zipcode | int(11) | NO |  | NULL |  |
| s\_branch | varchar(5) | NO |  | NULL |  |
| sex | varchar(6) | NO |  | NULL |  |
| dob | date | NO |  | NULL |  |
| status | int(11) | YES |  | NULL |  |

**Reports table:-**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| Student\_id | varchar(15) | Yes |  | Null |  |
| reason\_text | text | Yes |  | Null |  |
| reported\_id | varchar(15) | Yes |  | Null |  |
| rid | int(11) | No | PRI | None |  |
| subject | varchar(50) | Yes |  | Null |  |

**History table:-**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| id | int(20) | No | PRI | None |  |
| user\_id | varchar(20) | No |  | None |  |
| action | varchar(20) | No |  | None |  |
| loginif | datetime | No |  | None |  |

**Messages table:-**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| msg\_id | int(11) | No | PRI | None |  |
| reciver\_id | varchar(15) | No | PRI |  |  |
| sender\_id | varchar(15) | No | PRI |  |  |
| msg\_read | tinyint(1) | Yes |  | 0 |  |
| msg\_sub | varchar(100) | Yes |  | Null |  |
| Msg\_time | timestamp | No |  | current\_timestamp() |  |
| msg\_body | blob | Yes |  | Null |  |

**Contacts table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| user\_id | varchar(15) | No | PRI |  |  |
| contact\_id | varchar(15) | No | PRI |  |  |

**Chat table:-**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| sender\_id | varchar(15) | No | PRI |  |  |
| reciver\_id | varchar(15) | No | PRI |  |  |
| msg | varchar(200) | Yes |  | Null |  |
| msgtime | timestamp | No |  | current\_timestamp |  |

**4.2 Constraints on Tables**

1. Administrator is given username and password.
2. Existing database changes permission given only for administrator.
3. All the users are given permission for superficial access of their own data.
4. The data about each student should be consistent without any redundancy and can be accessed by the administrator.

|  |
| --- |
| **4.3 Front end implementation** |
|  |

**4.3.1 HTML**

## What is an HTML File?

* HTML stands for **H**yper **T**ext **M**arkup **L**anguage.
* Hyper text is ordinary text that has been dressed up with features such as formatting, images, multimedia and links to other documents.
* Markup is the process of taking ordinary text and adding extra symbols.
* An HTML file is a text file containing small **markup tags.**
* The markup tags tell the Web browser **how to display** the page.
* An HTML file must have an .**htm** or .**html** file extension.
* An HTML file can be created using a **simple text editor.**

## AN EXAMPLE OF HTML:

If you are running Windows, start Notepad and type in the following text:

<html>

<head>

<title> Title of page </title>

</head>

<body>

This is my first Home page <br> This is the bold </b>

</body>

</html>

## **4.4 Back end implementation**

## **4.4.1 PHP**

## What is PHP?

* PHP stands for **P**HP: **H**ypertext **P**reprocessor.
* PHP is a server-side scripting language, like ASP.
* PHP scripts are executed on the server.
* PHP supports many databases (MySQL, Informix, Oracle, Sybase, Solid, Generic ODBC, etc.).
* PHP is an open source software (OSS).
* PHP is free to download and use.
* PHP is a reflective programming language originally designed for producing dynamic web pages.

## Basic PHP Syntax:

A PHP scripting block always starts with **<?php** and ends with **?>**. A PHP scripting block can be placed anywhere in the document.

A PHP file normally contains HTML tags, just like an HTML file, and some PHP scripting code.

Below, we have an example of a simple PHP script which sends the text "Hello World" to the browser:

|  |
| --- |
| <html>  <body>  <?php  echo "Hello World";  ?>  </body>  </html> |

Each code line in PHP must end with a semicolon. The semicolon is a separator and is used to distinguish one set of instructions from another.

There are two basic statements to output text with PHP: **echo** and **print**. In the example above we have used the echo statement to output the text "Hello World".

**4.4.2 APACHE**

Apache is the most popular web server on the internet. Apache is an open source project like LINUX, MySQL and PHP. Apache is based on the NCSA (National Center for Super Computing Applications) web server.

In 1995-1996 a group of developers coalesced around a collection of patches to the original NCSA web server. This group evolved into the Apache has become a robust well documented multi-threaded web server. Particularly appealing in the 2.0 release improved support for non-UNIX systems.

Apache can run on a large number of hardware and software platforms. Since 1996 Apaches has been most popular web server on the internet. Presently apache holds 67% of the market.

## **4.4.3 MYSQL**

### What is MySQL?

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by MySQL AB.

**STATEMENTS AND FUNCTIONS:**

* Full operator and function support in the SELECT list and WHERE clause of queries.
* For example:
* mysql> **SELECT CONCAT(first\_name, ' ', last\_name)**
* -> **FROM citizen**
* -> **WHERE income/dependents > 10000 AND age > 30;**
* Full support for SQL GROUP BY and ORDER BY clauses. Support for group functions ([COUNT()](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_count), [COUNT(DISTINCT ...)](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_count), [AVG()](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_avg), [STD()](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_std), [SUM()](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_sum), [MAX()](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_max), [MIN()](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_min), and [GROUP\_CONCAT()](http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_group-concat)).
* Support for LEFT OUTER JOIN and RIGHT OUTER JOIN with both standard SQL and ODBC syntax.
* DELETE, INSERT, REPLACE, and UPDATE return the number of rows that were changed (affected). It is possible to return the number of rows matched instead by setting a flag when connecting to the server.
* The MySQL-specific SHOW statement can be used to retrieve information about databases, storage engines, tables, and indexes.
* The EXPLAIN statement can be used to determine how the optimizer resolves a query.

**4.5 SQL Queries:-**

**Create Queries:-**

**CREATE TABLE USERS(**

USER\_ID VARCHAR(15) NOT NULL,

PASSWORD VARCHAR(15) NOT NULL,

FNAME VARCHAR(15) NOT NULL,

LNAME VARCHAR(15) NOT NULL,

ADDRESS VARCHAR(30),

STATE VARCHAR(20),

COUNTRY VARCHAR(20),

ZIPCODE INTEGER DEFAULT 0,

S\_BRANCH VARCHAR(5) NOT NULL,

SEX VARCHAR(6) NOT NULL,

DOB DATE NOT NULL,

STATUS INTEGER DEFAULT 0,

PRIMARY KEY (USER\_ID));

**Insert Queries:-**

INSERT INTO `users` VALUES ('Dhanu','12345',’Dhanush','Gowda','mandya','karnataka','ind',56465,'cse','','1999-01-01',0),('Beeru','852147',’Beeresh','M P','mysore','karnataka','ind',571121,'cse','','1999-08-03',1);

**Display Queries:-**

Select \* from users;

**Update Queries:-**

update users set $field='$field\_value' where user\_id='$user';

**Search Queries:-**

select username, password from admin where username='$username' and password='$pass'"

select user\_id from users where user\_id ='$toaddr'

select user\_id from users where user\_id ='$fromaddr'

4.6 **TRIGGER** :

The MySQL trigger is a database object that is associated with a table it will be activated when a defined action is executed for the table the trigger can be executed when you run one of the following MySQL statements on the table that is INSERT, UPDATE and DELETE before or after the event.

CREATE TRIGGER `Insertuser` AFTER INSERT ON `users` FOR EACH ROW insert into history values(null,NEW.userid,’Inserted’,NOW());

|  |
| --- |
| CREATE TRIGGER `Deleteuser` AFTER INSERT ON `users` FOR EACH ROW insert into history values(null,OLD.userid,’Deleted’,NOW()); |

CREATE TRIGGER `Updateuser` AFTER INSERT ON `users` FOR EACH ROW insert into history values(null,NEW.userid,’Updated’,NOW());

4.6 STORED PROCEDURE :

In a **DBMS**, a **stored procedure** is a set of **SQL** statements with an assigned name that's **stored** in the **database** in compiled form so that it can be shared by a number of programs. Stored Procedure are Stored Programs, a program/function stored into database .

CREATE PROCEDURE ‘username’() NOT DETERMINISTIC NO SQL SQL SECURITY DEFINER select \* from user;

To execute the procedure, CALL ‘username’

**4.7 Pseudo Codes**

**4.7.1 HTML CODE**

**Index.html**

<html>

<head>

<title>

login page

</title>

<frameset rows="13%,\*" >

<frame src="main\_top.html" name="a" noresize>

<frame src="user\_admin.html" name="b" noresize>

</frameset>

</head>

<body>

</body>

</html>

**main\_top.html**

<html>

<head>

</head>

<body bgcolor="white">

<marquee BEHAVIOR=ALTERNATE scrollamount="5"><img src="img/Student\_Messengr.png" width=412 height=68></font> </marquee>

<a href="developers.html" target="\_top">Developers Edge</a>

</body>

</html>

**user\_admin.html**

<html>

<body background="img/bg.png">

<br>

<br>

<br>

<center><h2><a href="user\_index.html" target="\_top">users</a>

<br>

<a href="admin\_index.html" target="\_top">administrator</a>

<center></h2>

</body>

</html>

**main\_login.html**

<head>

</head>

<body background="wall.PNG"><center>

<font color="white">

<br>

<table>

<form method="get" action="login\_check.php"target="\_top">

<tr><td>User Name:</td><td> <input type="text" name="username"></td></tr>

<tr><td>Password:</td><td> <input type="password" name="password" ></td></tr>

<tr></tr>

<tr></tr>

<tr colspan=2><td></td><td><center><input type="submit" value="submit"></center></td></tr>

</form>

</table>

<?php

echo $\_POST[username];

echo $\_POST[password];

?>

Don't have a user id?

<a href="register.html" target="\_top">register now</a>

<p>Forgot password???

<a href="forgot\_password.html" target="b">click here</a>

</font>

</center>

</body>

</html>

**home\_tasks.html**

<html>

<head>

</head>

<body background="img/frame.png" link="black">

<div align=left>

<a href="home\_welcome.php" target="c">My Home</a><br><br>

<a href="compose\_mail.html" target="c">Compose mail</a><br><br>

<a href="inbox.php" target="c">Inbox</a><br><br>

<a href="sent\_box.php" target="c">Sent box</a><br><br>

<a href="contacts.php" target="c">Contacts</a><br><br>

<a href="online\_cht.php" target="c">Online Chat</a><br><br>

<a href="forum.html" target="\_top">Forums</a><br><br>

<a href="reports.html" target="c">Report an user</a><br><br>

</div>

</body>

</html

**4.7.2 PHP CODES**

**login\_check.php**

<?php

$servername='localhost'; // hostname or ip of server

$mysql\_login="a310"; // username and password to log onto db server

$mysql\_password="a310";

$dbname='a310'; // name of database

function connecttodb($servername,$dbname,$mysql\_login,$mysql\_password)

{

global $link;

$link=mysql\_connect ("$servername","$mysql\_login","$mysql\_password");

if(!$link){die("Could not connect to MySQL");}

mysql\_select\_db("$dbname",$link) or die ("could not open db".mysql\_error());

}

connecttodb($servername,$dbname,$mysql\_login,$mysql\_password);

session\_start();

$username=$\_REQUEST['username'];

$pass=$\_REQUEST['password'];

$res=mysql\_query("select user\_id, password from users where user\_id='$username' and password='$pass'");

$rows=mysql\_num\_rows($res);

if($rows>0)

{

?>

<script>

document.location="home.html";

</script>

<?php

$\_SESSION['username']=$user=$\_REQUEST['username'];

$password=$\_REQUEST['password'];

mysql\_query("UPDATE users SET status=1 WHERE user\_id='$user'");

}

else

{

?>

<script>

alert("invalid username or password. Please enter again");

document.location="index.html";

</script>

<?php

}

mysql\_close($con);

?>

**top.php**

<html>

<head>

</head>

<body bgcolor="white">

<marquee BEHAVIOR=ALTERNATE scrollamount="5"><img src="img/Student\_Messengr.png" width=412 height=68></font> </marquee>

<table width=100%>

<tr>

<td><a href="developers.html" target="\_top">Developers Edge</a> </td><td></td><td></td><td></td><td></td>

<td><div align="right"><a href="home.html" target="\_top"><?PHP session\_start(); Echo $\_SESSION['username']; ?></a>

<a href="logout.php" target="\_top">logout</a>

<a href="view\_profile.php" target="\_top">view profile</a></div></td>

</tr>

</table>

</body>

</html>

**home\_welcome.php**

<html>

<head>

</head>

<body>

<br>

<br><h1>Hi

<?php

session\_start();

echo$\_SESSION['username'];

?><img src="img/smiley-face-flat.png" width=50 height=50>

</h1>

<h2>Welcome to student messenger.</h2><h3>Click one of the links from the left pane to perform tasks.</h3>

</body>

</html>

**CHAPTER 5**

**TESTING**

**5.1 TEST CASE**

Test case is a set of test inputs, executions and expected results developed for a particular objective.

An excellent test case satisfies the following criteria:

* Reasonable probability of catching errors.
* Does interesting things.
* Doesn’t do unnecessary things.
* Neither too simple nor too complex.
* Allows isolation and identification of errors.

**5.2 CONCLUSION**

The project “**STUDENT MESSENGER**” meets all the requirements of the communication for all students in a college campus. It meets as a centralized system; it provides student, friends, online friends, discussed questions in forums details.

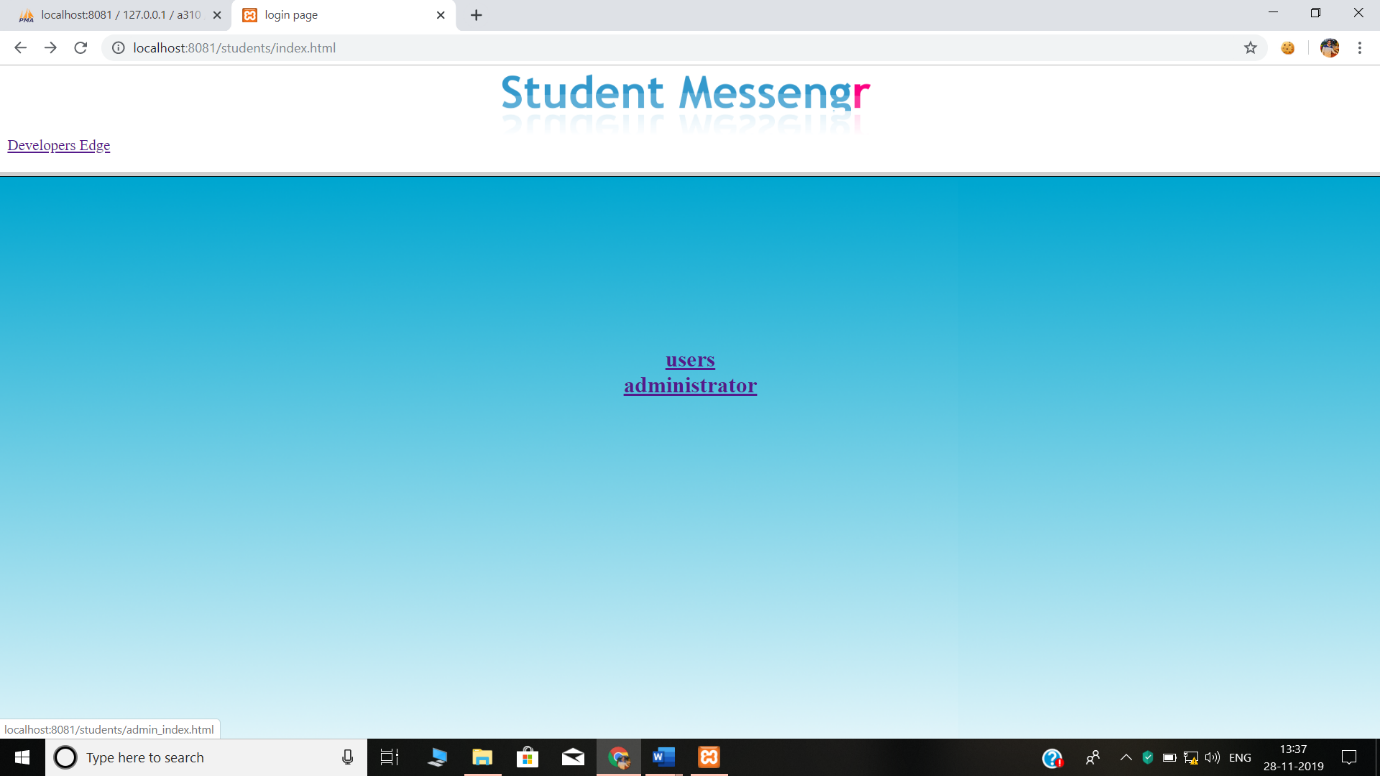
The following functions are implemented successfully for this project.

* Retrieving, modifying student information.
* Retrieving, modifying the information of the student registered.
* Successful maintenance of online chat.
* Obtaining list of online friends.
* Successful maintenance of the permissions used.
* Successful maintenance of the list of infrastructure.
* Successful maintenance of the users by administrator.
* Provides high security for users by the concept “sessions”.

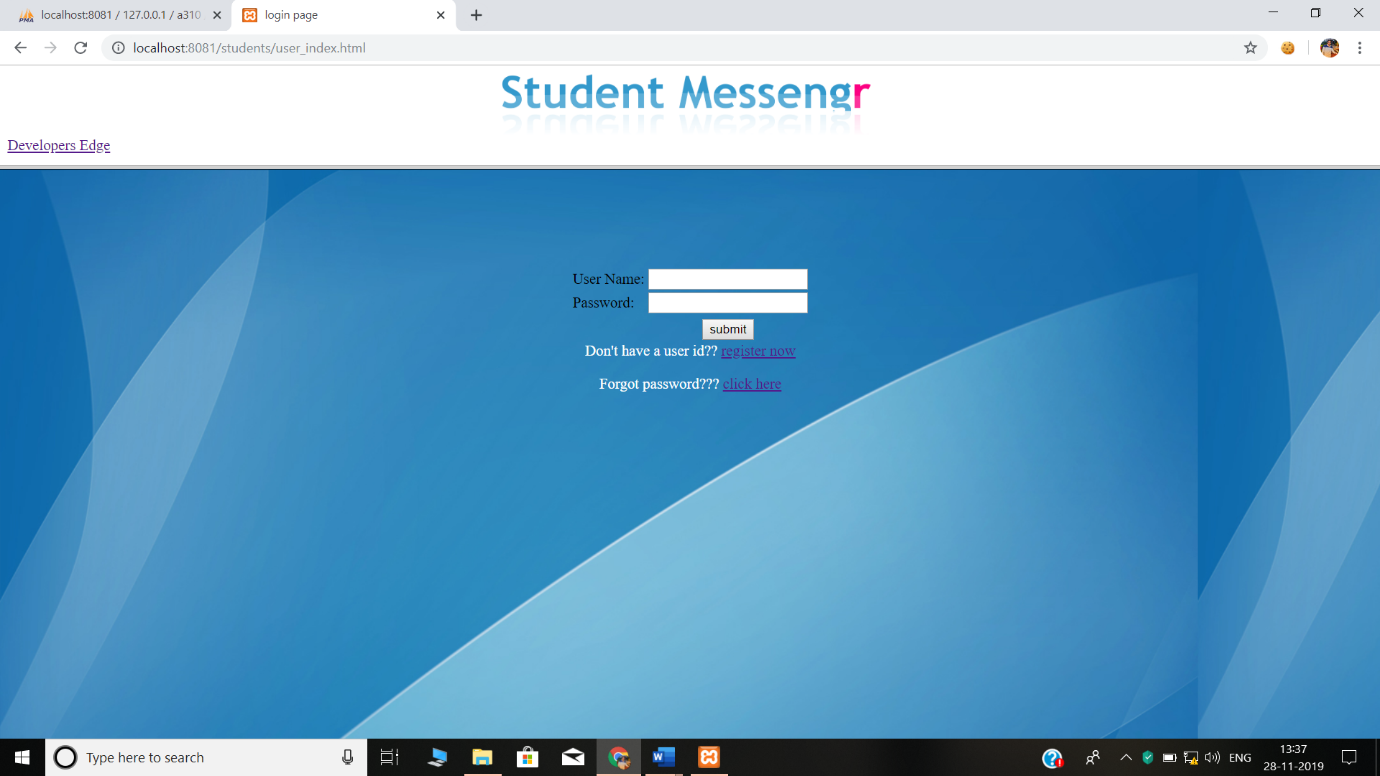
The future enhancement of this project is that this can be made online so that the students can use this STUDENT CONNECT from anywhere in the world through internet.

**5.3 SNAPSHOTS**

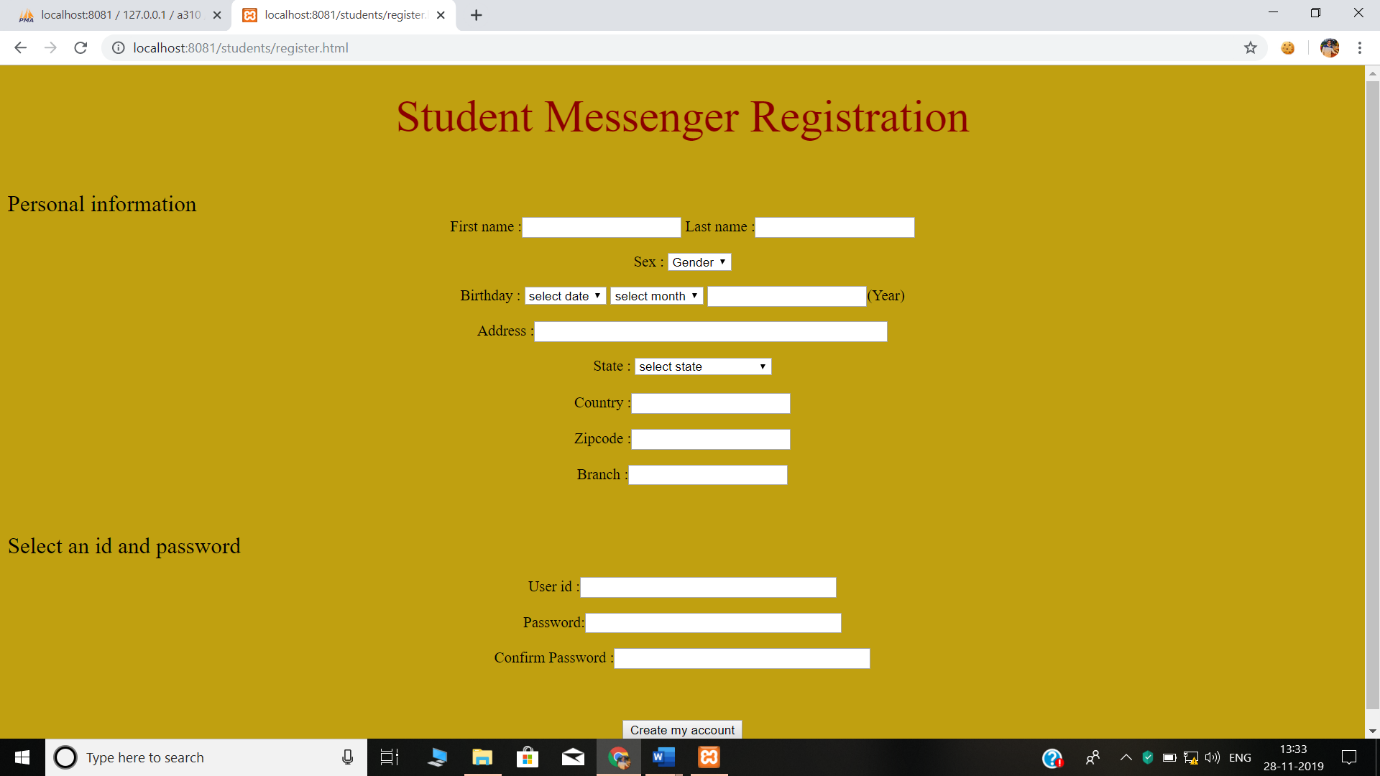
**Index page**



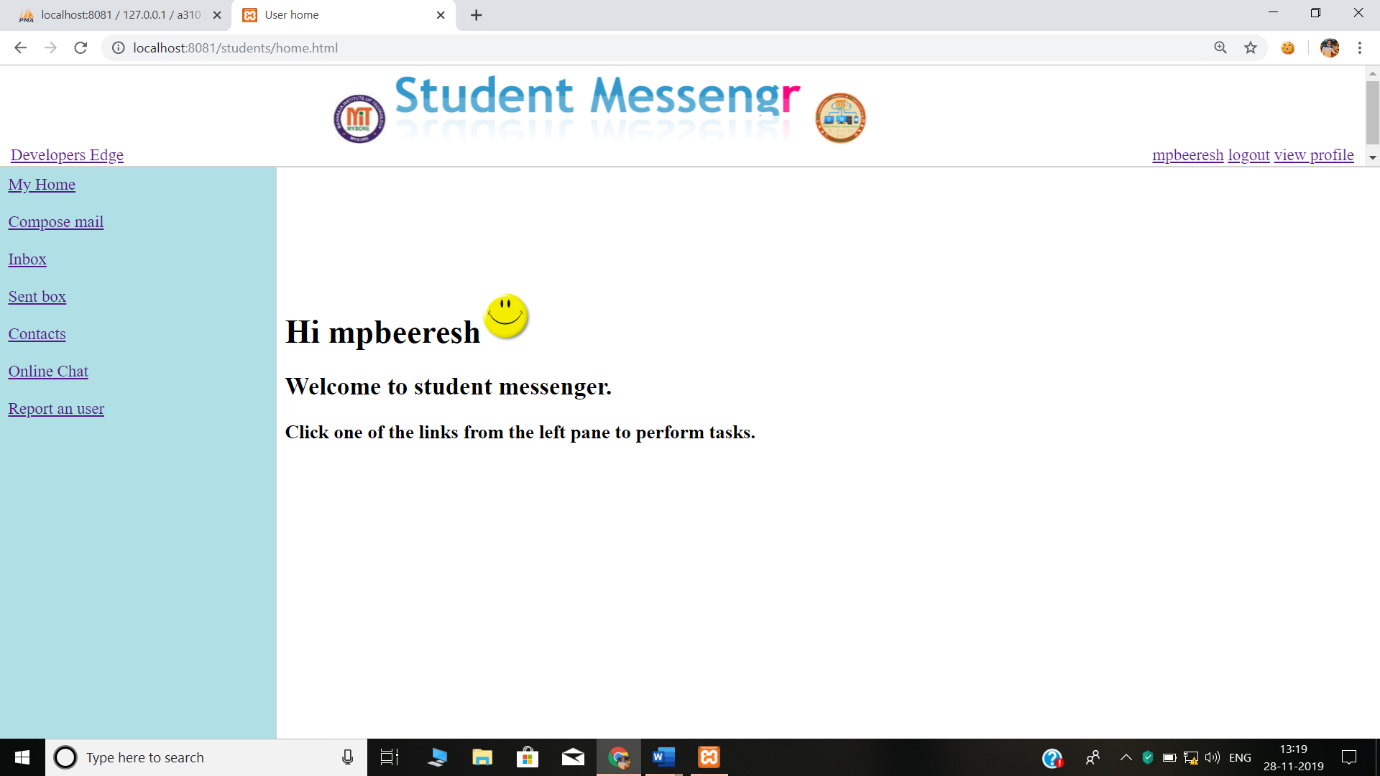
**User login**



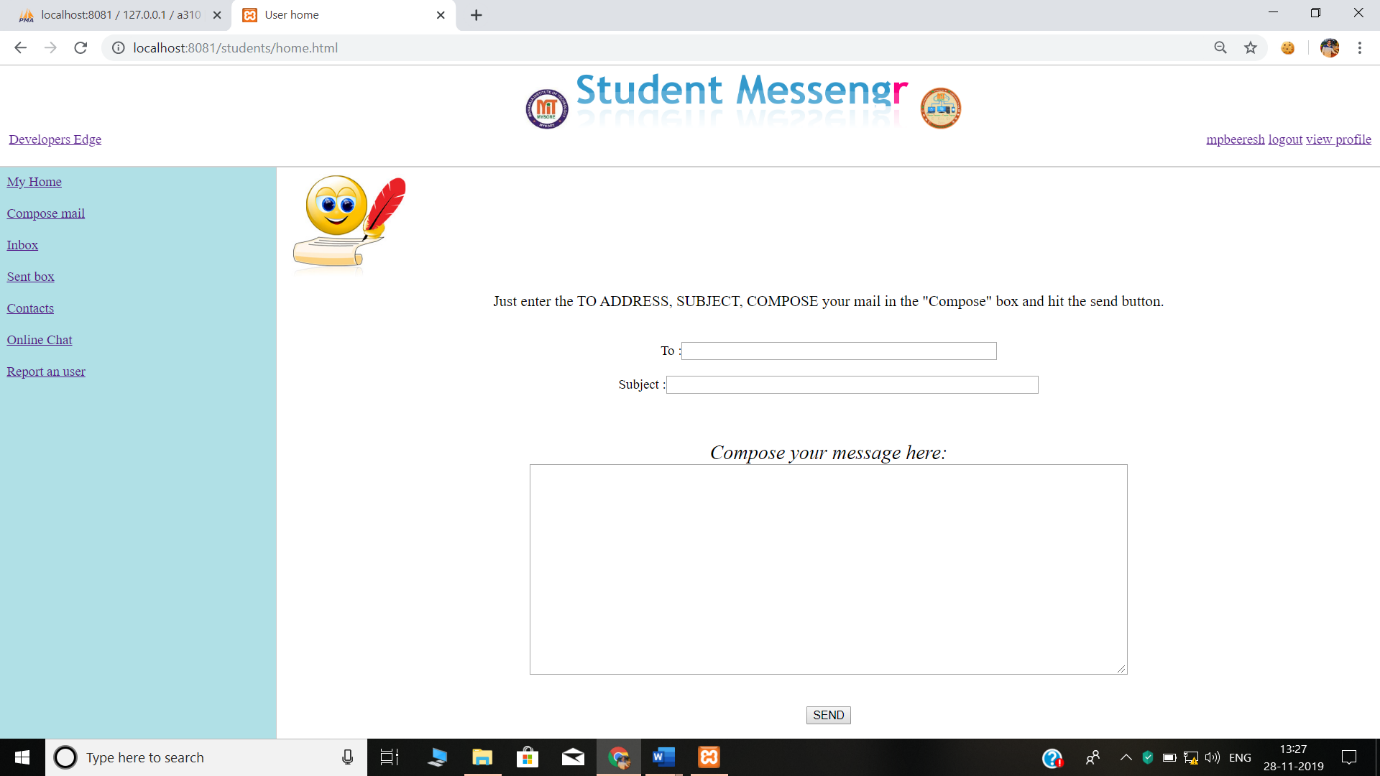
**Registration**



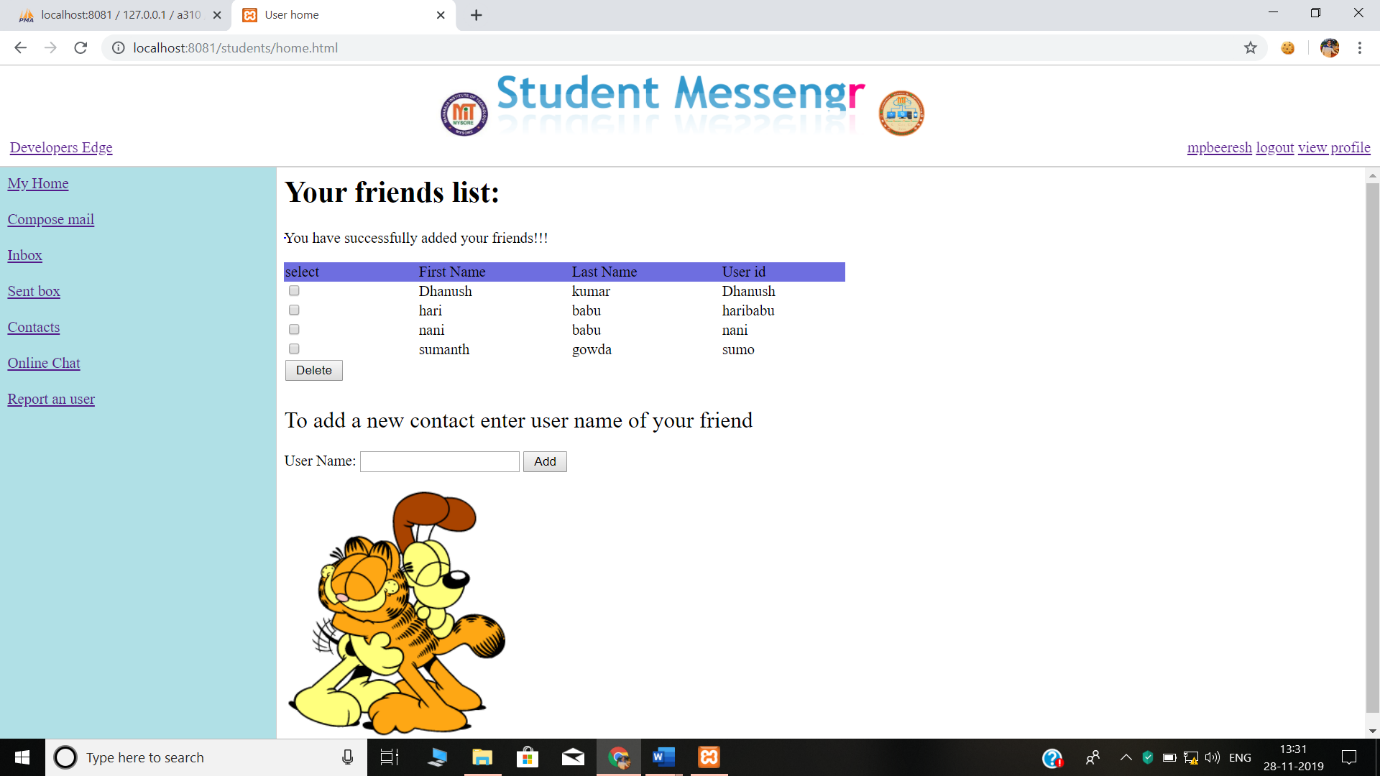
**MY HOME PAGE**



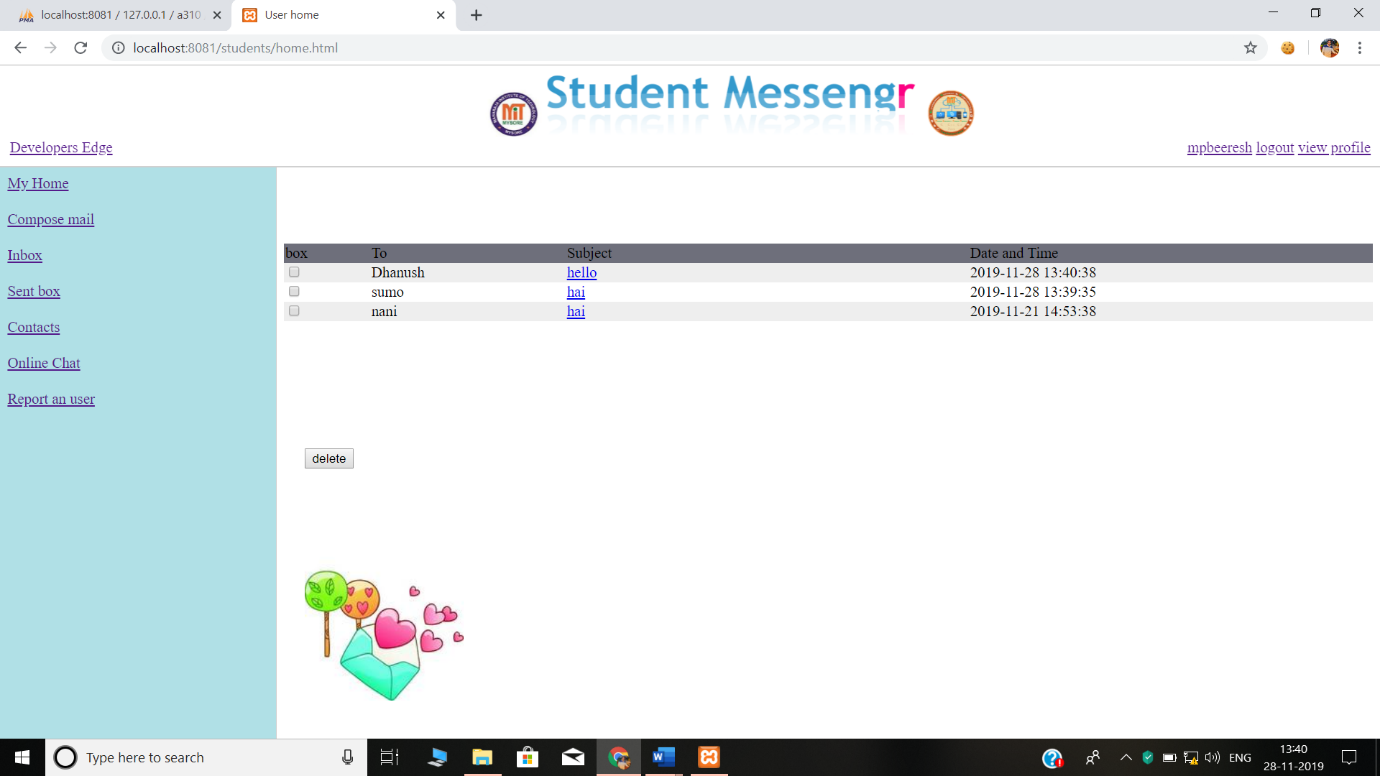
**Compose mail**



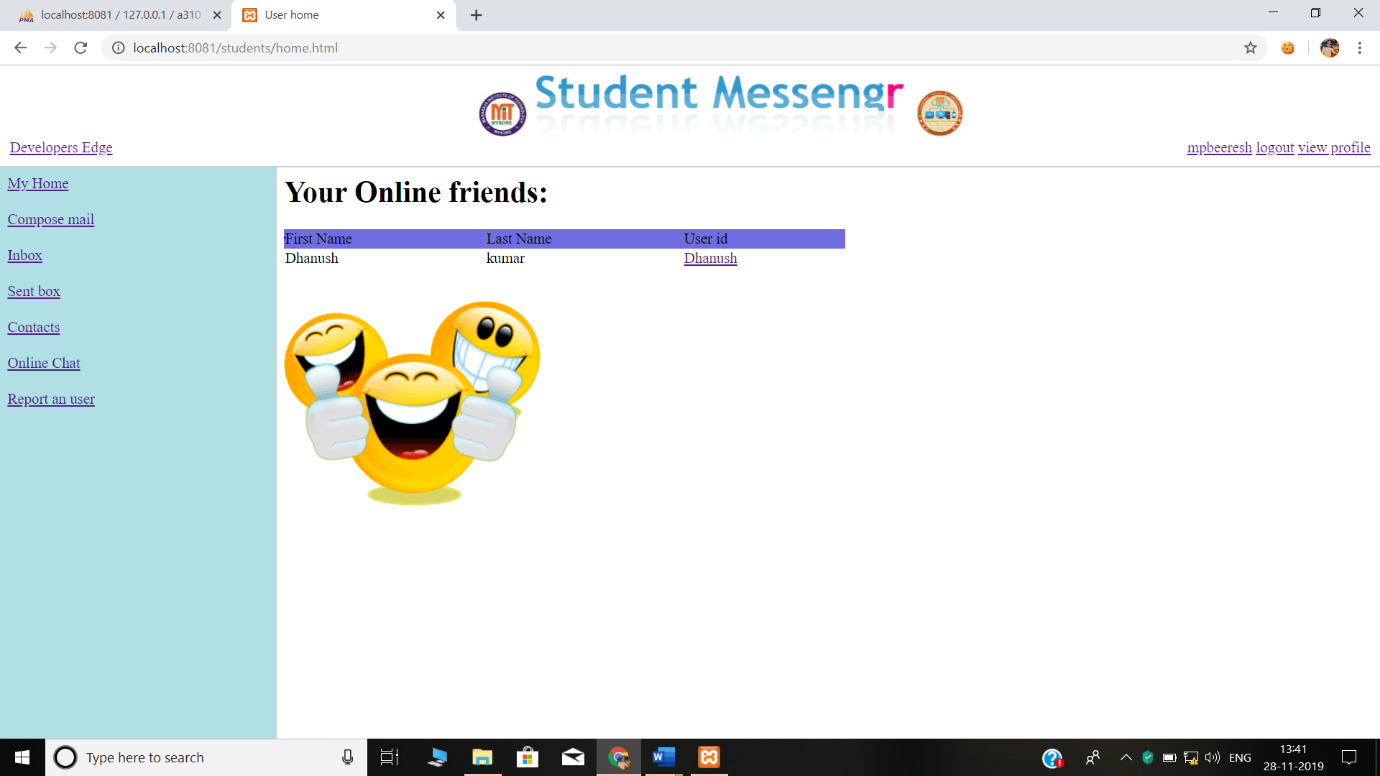
**Contacts**



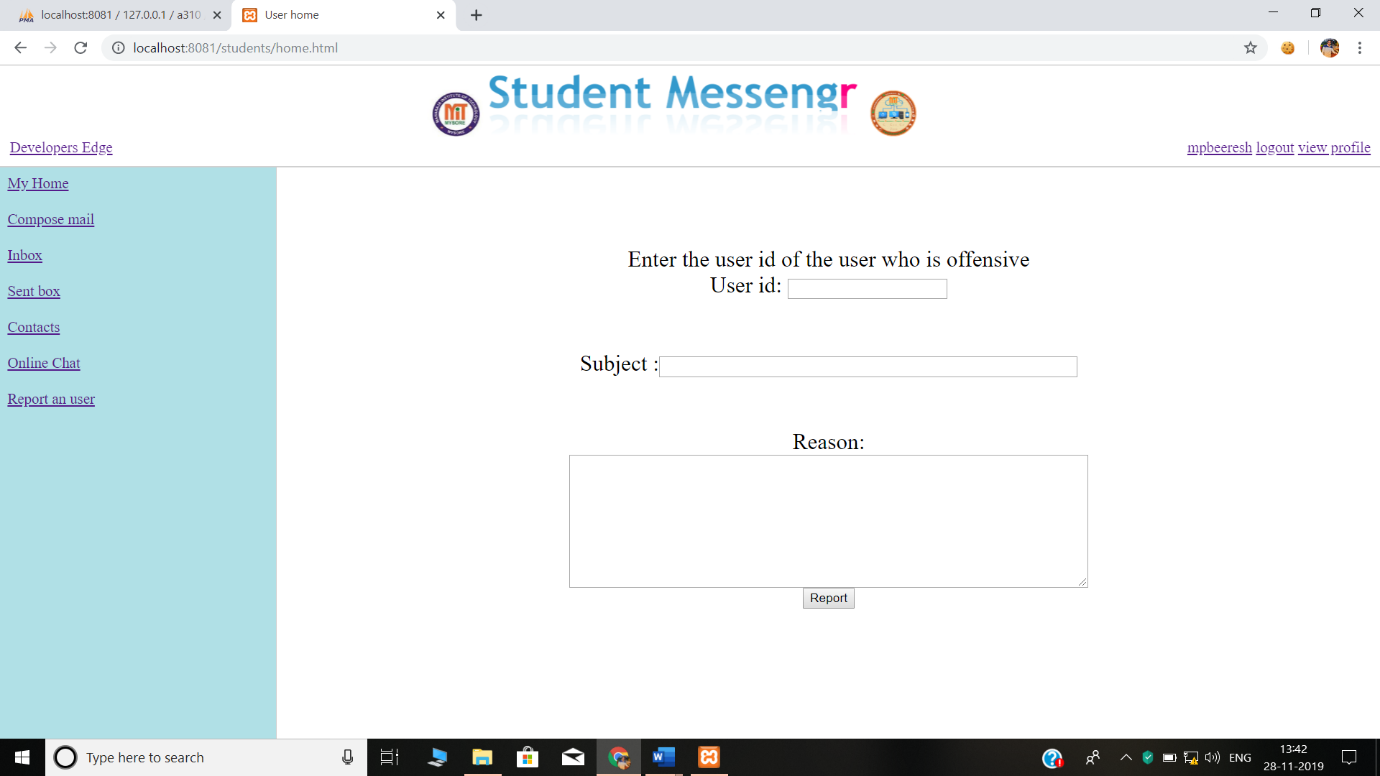
**SEND BOX**



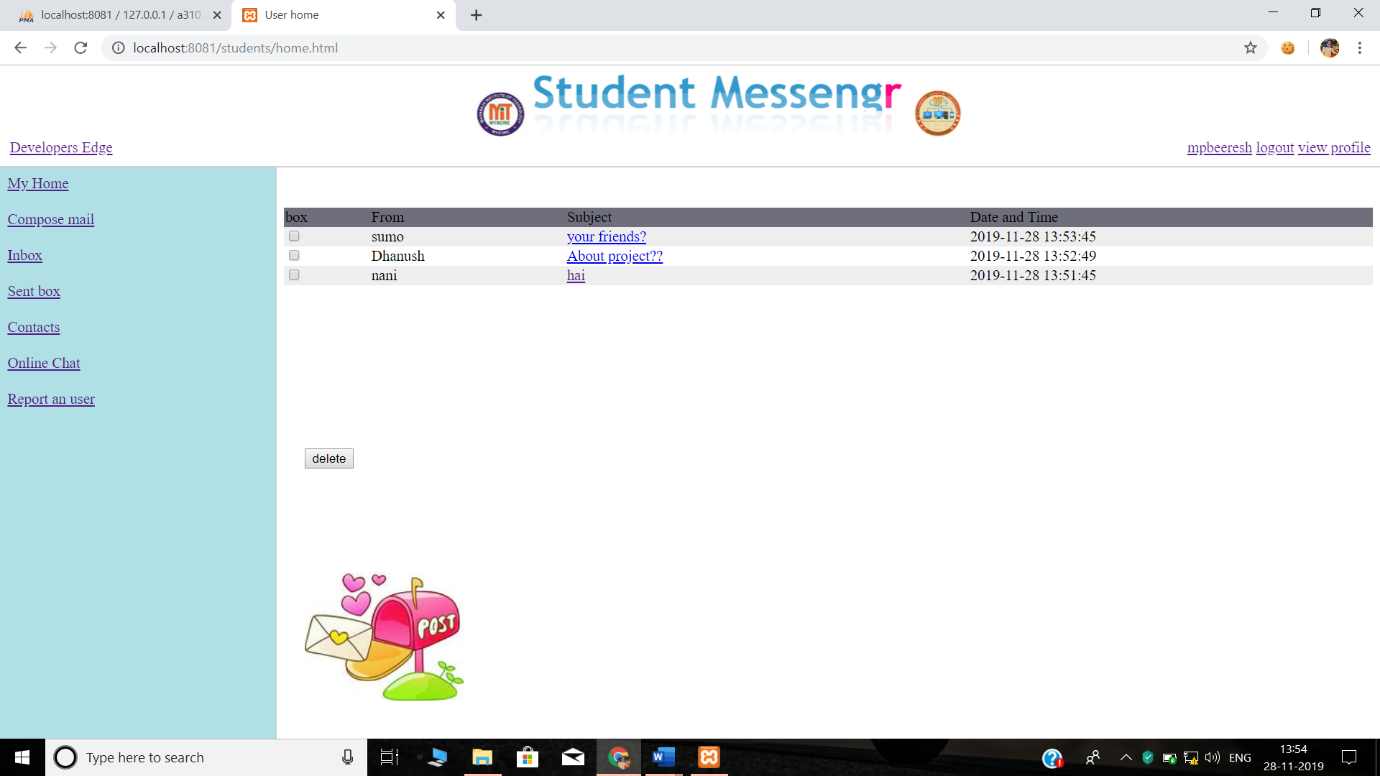
**ONLINE FRIENDS**



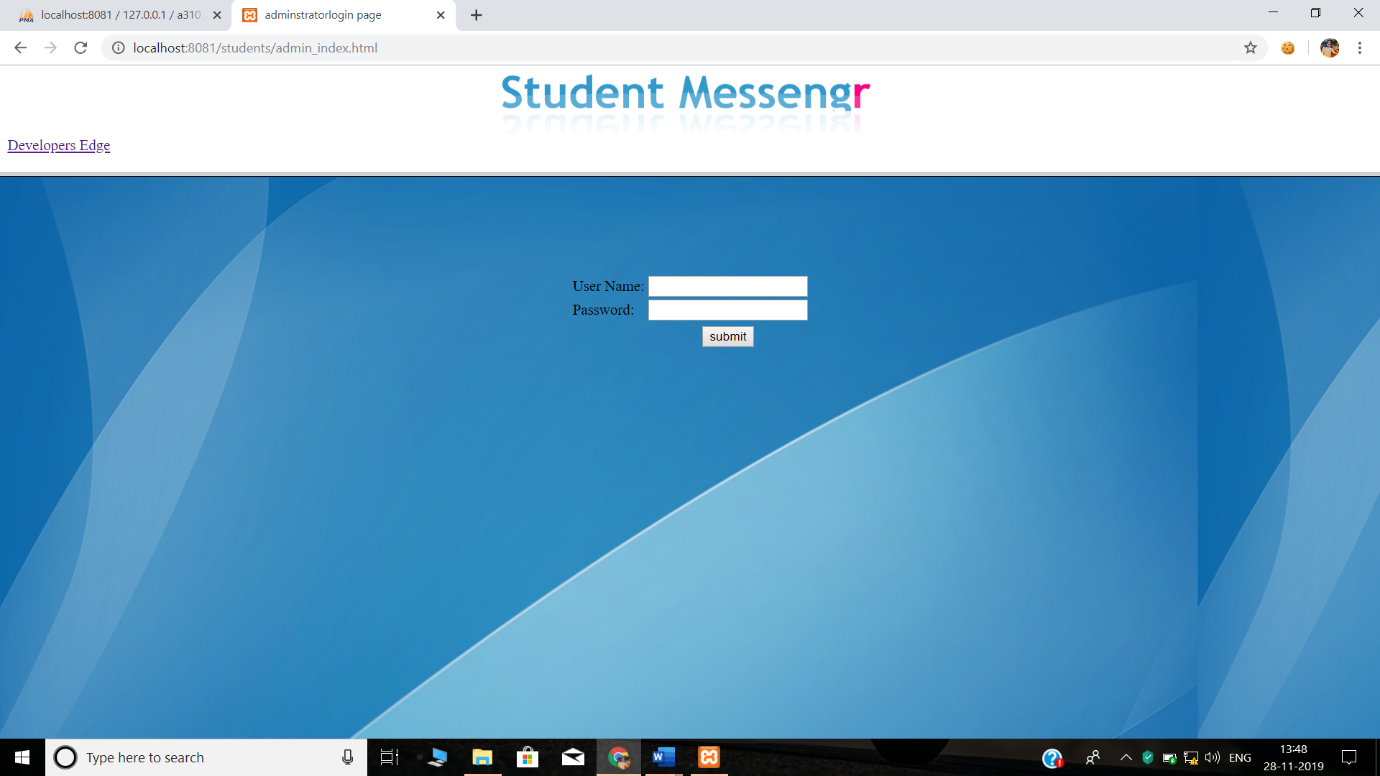
**REPORT AN USER**



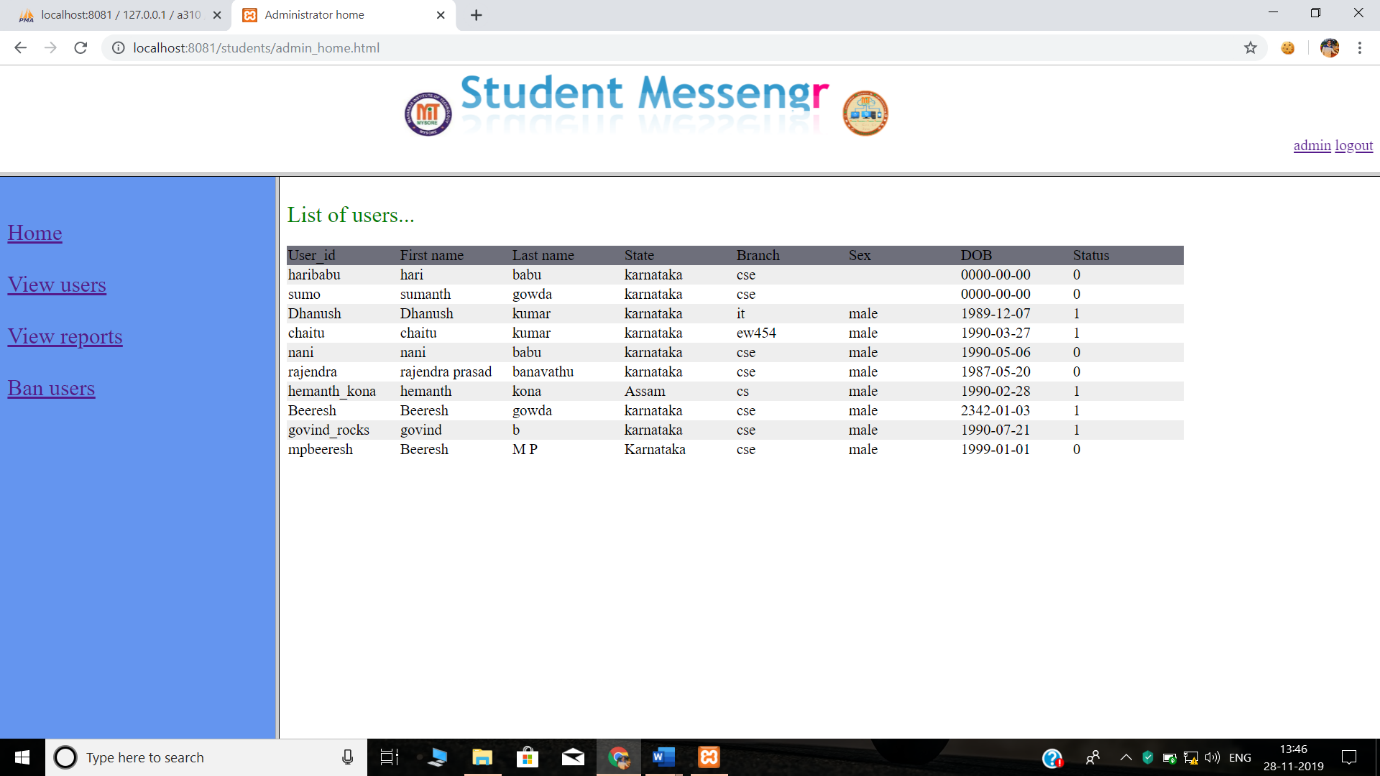
**INBOX**



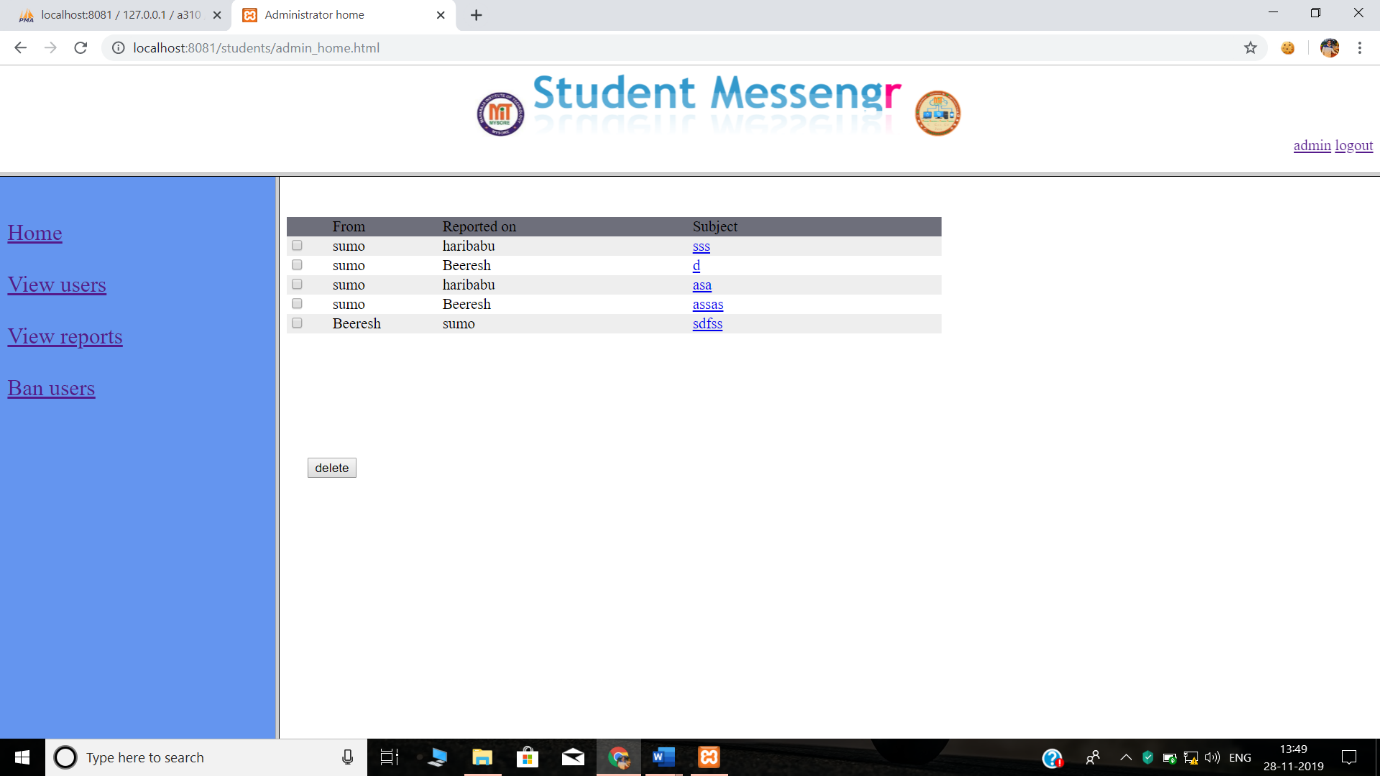
**ADMIN LOGIN**



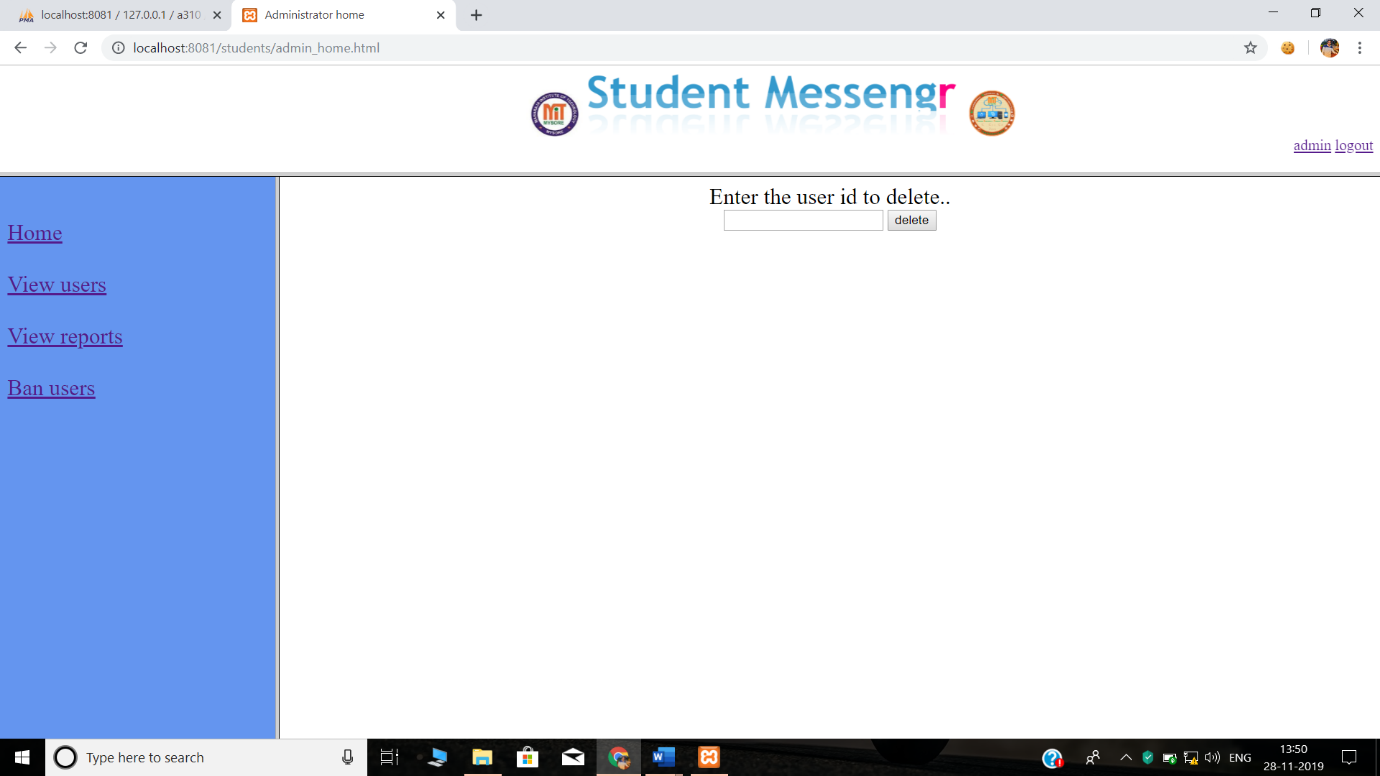
**VIEW USERS**



**VIEW REPORTS**



**BAN USERS**



|  |  |
| --- | --- |
| **REFERENCES** |  |

1. HTML BLACK BOOK “STEVEN HOLZNER”
2. “DATABASE MANAGEMENT SYSTEMS” Raghu Rama Krishnan Gehrke, McGraw-Hill Publications 3rd edition.
3. “DATABASE SYSTEM CONCEPTS” 4th Edition, by Silberschatz, Korth, Sudarshan .,McGraw-Hill Publications
4. Mysql Reference manual released by MySQLLAB corporation.
5. PHP PROGRAMMING “Vikram Vaswani”
6. PHP reference manual obtained from [www.php.net](http://www.php.net)
7. HTML and PHP reference from [www.w3schools.com](http://www.w2schools.com)

Youtube Video Links:-

* <https://youtu.be/A0vvNPDmp1U>
* <https://youtu.be/dwVj_g3TpZ4>
* <https://youtu.be/L5RpqspNAuc>